ResearchGate

See discussions, stats, and author profiles for this publication at: https://www.researchgate.net/publication/365368984

SIM-P - A Simplified Consensus Protocols Simulator: Applications to Proof of Reputation-X and Proof of Contribution

Article *in* IEEE Internet of Things Journal · November 2022

DOI: 10.1109/JIOT.2022.3221916

CITATIONS
0

READS
146

4 authors:



Damilare Peter Oyinloye
Kwara State University
5 PUBLICATIONS **82** CITATIONS
SEE PROFILE



Je Sen Teh
Universiti Sains Malaysia
52 PUBLICATIONS **1,316** CITATIONS
SEE PROFILE



Norziana Jamil
Universiti Tenaga Nasional (UNITEN)
125 PUBLICATIONS **852** CITATIONS
SEE PROFILE



Jiashen Teh
Universiti Sains Malaysia
95 PUBLICATIONS **2,375** CITATIONS
SEE PROFILE

All content following this page was uploaded by Je Sen Teh on 14 November 2022.

The user has requested enhancement of the downloaded file.

SIM-P – A Simplified Consensus Protocols Simulator: Applications to Proof of Reputation-X and Proof of Contribution

Damilare Peter Oyinloye*, Je Sen Teh*, Norziana Jamil and Jiashen Teh

***Abstract*—Blockchain is a distributed ledger in which participating users with varying levels of trust agree on the ledger's content using a consensus mechanism called consensus protocols. There has been a rising interest in the design of consensus protocols since they play a central role in blockchain architecture. However, many recently proposed consensus protocols lack experimental verification which hampers the possible deployment of these protocols in real-world blockchain networks. In this paper, we propose a simple tool called SIM-P that can accurately simulate the behavior of these consensus protocols with ease. It is an agent-based stochastic simulator that relies on the sequential Monte Carlo method to model how block publishers are selected. The likelihood of each node (represented as agents) being selected as a block publisher is represented by independent trials in a binomial experiment. We provide a base SIM-P model that simulates Proof of Work (PoW) for benchmarking purposes. The PoW model also serves as the basic structure of the simulator that can be adapted to other protocols. We showcase the flexibility of SIM-P by proposing two additional simulation models for– Proof of Reputation- X and Proof of Contribution, both of which lack experimental verification in their original design specifications. We show how the simulator can be used to produce vital metrics such as throughput, resistance against the 51% attack, and energy consumption. We verify the accuracy of SIM-P by comparing PoW's simulated results with theoretical estimates and historical Bitcoin data.**

***Index Terms*—Blockchain, simulator, consensus protocol, Proof of Work, Proof of Reputation, Proof of Contribution**

I. Introduction

Blockchain technology has seen a meteoric rise in popularity since its inception. Cryptocurrencies, arguably its most well-known application, facilitate safe (and anonymous) financial transactions between various parties. Blockchain has been applied in other various areas including the *Internet of Things* (IoT) [1] [2]. For example, it has seen use in privacy and energy management of IoT devices [2] and authentication for smart farming [3]. Blockchain can be likened to a fully replicated distributed database system that records all network transactions. Every full node within the network stores a copy of the blockchain along with transaction information while partial nodes (or lightweight nodes) only store parts of the blockchain required for verification purposes. As opposed to a conventional centralized ledger hosted by a single point of authority, the blockchain is thus collaboratively managed by multiple participating nodes. This collaborative environment requires a consensus or decision-making mechanism that allows nodes to agree upon transactions that will be included in the blockchain. Therefore, a vital architectural component of the blockchain is its underlying consensus mechanism, referred to as the consensus protocol. In addition, these protocols also usually decide on which node will be given the authority to add a new (data) block to the chain.

Consensus protocols have been existing before the emergence of blockchain technology. They have been used in distributed computing environments to ensure overall system reliability. In the blockchain context, consensus protocols are, at its core, a set of regulations used by nodes in a blockchain network to determine if a transaction is valid. This provides assurance to all parties that they hold the same, up-to-date transaction ledger. Blockchain consensus protocols can be classified into two distinct classes: (1) proof-based consensus protocols, which require that nodes provide proof of effort or resource expenditure, and (2) voting-based consensus protocols, which require participating nodes in the network to exchange results of verifying new blocks or transactions before making the final decision about which node is allowed to commit a new block to the chain. Well-established consensus protocols include Proof of Work (PoW) [4], Proof of Stake (PoS), [5] and their variants [6].

Numerous lesser-known alternative blockchain consensus protocols have been proposed over the years, many of which were designed for specific areas such as medicine [7], Internet-of-Things (IoT) [1] [2], Agriculture [3] or electric vehicles [8], [9]. Unfortunately, many of these protocols remain only of academic interest and have yet to see real-world applications [10]–[16]. However, they still possess unique features that may be useful for the advancement of blockchain technology [17]. These alternative consensus protocols can be broadly divided into several categories based on how block publishers (or block leaders) are selected, some of which are as follows:

- Consensus protocol based on Effort or Work (CPE)
- Consensus protocol based on Wealth or Resources (CPW)
- Consensus Protocol based on Past Behavior or Reputation (CPPB)
- Consensus Protocol based on Representation (CPR)

Many simulators have been proposed over the years for

The full authenticated version of this paper has been published in the IEEE Internet of Things journal and is available at https://doi.org/10.1109/JIOT.2022.3221916.

D. P. Oyinloye, J. S. Teh and J. Teh are with Universiti Sains Malaysia.

D. P. Oyinloye is also with the Department of Computer Science, Kwara State University, Nigeria.

N. Jamil is with Universiti Tenaga Nasional.

*Corresponding Authors (e-mail: damilare.oyinloye@student.usm.my, jesen_teh@usm.my)

blockchain networks that use popular protocols such as PoW and PoS [18]–[20]. Although these were blockchain simulators rather than simulators that catered specifically to consensus protocols, they still provided experimental data for these protocols in terms of performance and security. However, adapting some of these simulators to other protocols, especially alternative protocols, is non-trivial. Additionally, these simulators also include the other technology layers in the blockchain network which are not required if the goal is just to simulate a consensus protocol. Hence there is a need for a simple yet flexible consensus protocol simulator that can be adapted to various types of consensus protocols with ease.

To this end, we propose a simplified consensus protocol simulator (SIM-P) which is an agent-based stochastic simulation framework. SIM-P relies on the sequential Monte Carlo (MC) method to model how block publishers are selected, i.e., the likelihood of nodes (represented as agents) being selected as block publishers is represented by independent trials in a binomial experiment. We provide a *base model* based on PoW, on which we verify the accuracy of the proposed simulator using theoretical and real (historical) Bitcoin data. In addition, models for other protocols can be easily developed by simply modifying the base model and adding additional attributes. We also provide simulation models for two alternative protocols – Proof of Reputation-X (PoRX) [14] and Proof of Contribution [16] – to showcase the flexibility of SIM-P. As an additional contribution, we provide an experimental comparison of these protocols against PoW in terms of throughput, security against the 51% attack and energy consumption, all of which can be measured using SIM-P. A summary of our contributions is as follows:

1) A multi-agent-based, stochastic simulator for consensus protocols.
2) PoW simulation model as a *base model* or benchmark.
3) Proof of concept models for other consensus protocols such as PoRX and PoC.
4) Performance and security analysis of PoRX and PoC benchmarked against PoW.

The rest of this paper is organized as follows: Section II provides a brief look at related work, comparing the different types of blockchain consensus protocol simulators. Section III presents an overview of blockchain consensus protocols and their classifications followed by stochastic simulation and agent-based modeling. The architecture of SIM-P and its applications to protocols such as PoW, PoRX, and PoC are presented in Section IV. In Section V, we show how SIM-P can be used to obtain performance metrics for consensus protocols by performing a comparison between the base model (PoW) and other models (PoRX, PoC). The paper is concluded in Section VI.

II. Related Work

In this section, we take a look at some of the existing blockchain simulators. We focus only on open-source simulators that have been described in peer-reviewed academic publication venues. Alharby and van Moorsel proposed Blocksim, a tool designed to construct discrete-event system models for blockchain systems [18]. It supports the analysis of a large range of blockchain systems. The *base model* of the simulator (the main model) is made up of the common blockchain abstraction layers such as the network layer, consensus layer, and incentives layer. The base model can be used for a wide range of blockchain systems and can be easily extended to include other system or deployment requirements. The base model and other models of the Blocksim are simulated using Python. These models include Ethereum and other PoW-based blockchain models.

Faria and Correia proposed another discrete-event simulation framework, also called Blocksim, that helps in the design, implementation, and evaluation of both existing and new blockchain networks [19]. To avoid confusion with the earlier Blocksim tool, we refer to this one as Blocksim(F). It follows a stochastic simulation model which represents random phenomena by sampling from a probability distribution. It provides a framework and a set of base simulation models common to several blockchain systems. Blocksim(F) simulates fundamental blockchain layers such as the consensus and incentive layer but does not include the network layer. Users have the option of extending these models to evaluate their own design and implementation decisions. BlockSim(F) can be used to simulate Bitcoin and Ethereum.

As its name implies, Bitcoin Simulator [21] was designed to simulate the Bitcoin blockchain. Various protocol and network parameters can be specified by users such as the number of nodes and network bandwidth. The main goal of this simulator was to study the trade-off between performance and security. It simulates the execution of a blockchain network at the block level rather than the transaction level. Bitcoin Simulator does not provide fine-grained control over the application, limiting its applicability for broader adoption. It also inserts a series of static time stamps to simulate the PoW consensus protocol, which does not precisely characterize the behavior of real-world blockchain systems. For example, Bitcoin dynamically adjusts PoW difficulty, and nodes complete the tasks in stochastic time intervals. Also, Bitcoin Simulator is based on a discrete-event network simulator known as NS3 [22]. This, unfortunately, limits its scalability as it can only support up to 6000 nodes.

Vibes [20] is an extension of the Bitcoin Simulator with several improvements. VIBES supports a web-based interface for users to visually track the growth of the network. VIBES improves the scalability of the Bitcoin Simulator by employing a fast-forwarding algorithm that designates a coordinator to control events according to existing nodes' best guesses on the block creation time. Such a centralized coordinator is better suited for single node simulations at small or medium-scale but could be a performance bottleneck for extreme-scale applications. Similar to Bitcoin Simulator, VIBES takes the same approach of inserting timestamps to hypothetically carry out the PoW workload. Both Bitcoin Simulator and VIBES are coarse estimators of real-world blockchain executions due to the lack of real PoW implementations or a decentralized architecture.

Blockbench is a simulation framework for analyzing private blockchains with throughput, latency, scalability, and fault-

tolerance as performance parameters [23]. It currently supports Ethereum, Parity, HLF, and Quorum, which are four prominent private blockchain platforms, but claims to also be able to evaluate any private blockchain. It covers four layers of the blockchain network (consensus, data model, execution engine, and application).

BlockLite is a lightweight single-node emulator of blockchain systems [24]. BlockLite is designed to execute real PoW workload along with peer-to-peer network communications and hash-based immutability. It conducts the real PoW workload by actually computing and comparing block hash values against predefined thresholds. It employs a pre-processing approach to avoid the per-node computation overhead at run-time and thus scales to thousands of nodes. Moreover, BlockLite offers an easy-to-use programming interface allowing for a building block-like customization to the system.

SimBlock [25] is an event-driven blockchain network simulator. It treats the creation of blocks, and sending and receiving of messages as events. SimBlock, in contrast to other simulators, makes it easier to alter the node's behavior, making it possible to observe how the actions of the nodes affect the blockchain network. SimBlock does not perform the actual hash computation to determine the success of a node in a mining round. Rather, the sum of the block generating capacity of each node and the block generation difficulty determines when mining is successful. Also, the neighbor nodes selection technique can be implemented using SimBlock. Since the success probability of block generation is used to determine block creation time, it is unnecessary to replicate the mining process, which uses a lot of processing power. Instead, a network with numerous nodes can be simulated.

DAGsim is an asynchronous, continuous-time, and multi-agent simulation framework for DAG-based protocols [26]. It models honest and semi-honest participants in the network to analyze the behavior of the IOTA cryptocurrency. The simulation framework does not model malicious agents and has a high run time complexity. DAGsim was designed to facilitate the expansion of its feature set.

These simulators, including DAGsim which is also an agent-based simulator, are better suited for the design and analysis of the entire blockchain network rather than just individual protocols. They simulate a wide range of blockchain layers that are interdependent and also depend on a wide range of parameters. Modifying these simulators to specifically simulate a consensus protocol is not a trivial task since the various abstraction layers covered by these tools (summarized in Table I) need to be taken into consideration. Hence, SIM-P fills this gap by providing an open-source consensus protocol simulation tool that can be used to analyze protocols with ease.

Ideally, a consensus protocol simulator should be able to simulate several key metrics which include the number of blocks generated, average block discovery time, dynamic difficulty fluctuations, incentivization, block orphaning, throughput, and other security parameters. Table II summarizes the key features supported by each of these existing simulators, along with the proposed simulator, SIM-P.

III. Preliminaries

A. Consensus Protocols

A consensus protocol is the backbone of a blockchain network – enforcing integrity and consistency. It also contributes to tamper-proofing and the immutability of the blockchain. In general, these protocols allow nodes to collaboratively verify transactions and decide on who is allowed to append a new block to the chain while at the same time preventing malicious nodes from appending falsified data. These protocols can be broadly categorized into two groups: proof-based and voting-based protocols. Some of these protocols are well-known and referred to henceforth in this paper as *mainstream* consensus protocols. Several examples include PoW [4], PoS [5], and their variants [6]. In contrast, we refer to relatively new consensus protocols that rely on other types of mechanisms as *alternative* new consensus protocols. Some of these alternative protocols are designed for specialized applications but general-purpose variants also exist. Examples include PoRX [14], PoC [16], Proof of Familiarity (PoF) [27], Proof of Luck (PoL) [15], Proof of X-Repute (PoX-R) [28], and PoRX [14], Next, we provide the technical details for PoW, PoRX, and PoC since they will be the three protocols simulated in this work.

1) Proof of Work: PoW is arguably the most widely adopted consensus protocol since the introduction of Bitcoin in 2009 [29]. In PoW, nodes amass computational power to increase their chances of being selected to add new blocks to the blockchain and receive a *mining* reward [30]. Computational power is expended to solve a hash puzzle, the solution of which is included in the resulting block. Due to its stochastic nature, the process of solving this hash puzzle is computationally intensive, which causes PoW to consume an immense amount of energy [31]. Apart from wasteful energy consumption, other drawbacks of PoW include low throughput and possible decentralization due to mining pools.

Solving the aforementioned hash puzzle is, at its core, a brute force search for hash values that fulfill certain requirements. Difficulty D defines how difficult it is to find a hash value below a specific target T. For a node to be selected to publish a valid block, the node must produce a hash below T, which is a 256-bit number with leading zeros. With a higher target, nodes will find the desired hash value at a higher rate. Therefore, difficulty/target can be adjusted to control the network's throughput. The target of a node is calculated with respect to global difficulty as

$$T = \texttt{0xFFFF} \cdot \frac{2^{208}}{D}, \tag{1}$$

where the numerator $\texttt{0xFFFF} \cdot 2^{208}$ is hexadecimal value of the highest possible target.

2) Proof of Reputation-X: PoRX [14] integrates a reputation component into Po*X* protocols, where the *X* refers to the consensus mechanism such as *work* in PoW or *stake* in PoS. Nodes increase their likelihood of being selected to publish a block by accumulating an additional resource known as reputation through good conduct and behavior history. Reputation earned is stored in the reputation module that dynamically

TABLE I
COMPARISON OF CONSENSUS PROTOCOL SIMULATORS

Ref.	Simulator	Consensus Protocol	Consensus	Network	Data	Model Type	Framework Language
[21]	Bitcoin-Simulator	PoW	Yes	Yes	No	discrete-event	C++/NS3
[20]	VIBES	PoW	Yes	No	Yes	discrete-event	Scala
[19]	Blocksim (F)	PoW	Yes	Yes	Yes	discrete-event	Python/Simpy
[18]	Blocksim	PoW	Yes	No	Yes	discrete-event	Python
[23]	Blockbench	Permissioned	Yes	Yes	Yes	discrete-event	C++/Python/Java
[24]	BlockLite	PoW	Yes	Yes	Yes	discrete-event	Java
[25]	SimBlock	PoW	Yes	Yes	Yes	discrete-event	Java
[26]	DAGsim	DAG	Yes	Yes	No	agent-based	Python/Mesa

TABLE II
FEATURES OF A BLOCKCHAIN CONSENSUS PROTOCOL SIMULATOR. N DENOTES THE NUMBER OF BLOCKS GENERATED, BD DENOTES AVERAGE BLOCK DISCOVERY TIME, D DENOTES DYNAMIC DIFFICULTY ADJUSTMENT, I DENOTES INCENTIVIZATION, OB DENOTES THE NUMBER OF ORPHAN BLOCKS, TP DENOTES THROUGHPUT, AND SP DENOTES SECURITY PARAMETERS.

Simulator	N	BD	DA	I	OB	TP	SP
Bitcoin-Simulator	Y	Y	N	N	Y	N	Y
VIBES	N	Y	N	N	Y	Y	Y
Blocksim (F)	Y	N	N	N	N	N	N
Blocksim	Y	N	N	Y	Y	Y	N
Blockbench	Y	Y	N	Y	Y	Y	Y
SimBlock	Y	Y	N	N	Y	N	N
BlockLite	Y	Y	N	N	N	Y	N
DAGsim	Y	Y	N	N	N	N	N
SIM-P	**Y**	**Y**	**N**	**Y**	**Y**	**Y**	**Y**

adjusts puzzle-solving difficulty for individual nodes. The goal of PoRX is to eliminate the risk of centralization from the use of ASIC and mining pools. In PoRX, nodes with higher reputation have lower mining difficulty as compared to nodes with a lower reputation. After successfully finding a block, a node is rewarded with more reputation along with mining fees. However, if a node fails to produce a block within a stipulated time, it faces a reduction in reputation. Reputation also experiences a decay over time.

Let node reputation, maximum reputation, and global difficulty be denoted as R, R_m, and D respectively. Additionally, a node's *local* difficulty level after being influenced by its reputation is D_l and the conversion rate factor between reputation and difficulty is R_c. If R is less than or equal to R_m, the difficulty for the node is calculated as

$$D_l = D + \frac{R_m - R}{R_m} \cdot \frac{D}{R_c}, \tag{2}$$

but if R is greater than R_m, the new reputation influenced difficulty for the node is calculated as

$$D_l = D - \frac{R - R_m}{R_m} \cdot \frac{D}{R_c}. \tag{3}$$

The value of D_l of each node is then passed on to the underlying PoX module to adjust the node's likelihood of being selected as a block leader. For example, if the underlying protocol is PoW, D_l will be used to calculate the node's target in Eq. 1 instead of D.

At the end of each cycle, nodes are awarded reputation for contributing positively to the network or otherwise penalized. The reward and punishment module for PoRX are governed by the following rules:

- If a node wins the right to produce a block and the resulting block is successfully added to the blockchain with all of its transactions verified, the node receives both reputation and mining rewards (e.g. tokens).
- Nodes that fail to produce blocks within a specific time period or fail to produce an expected number of blocks will be penalized, i.e., have their reputation values deducted.
- If the current reputation value of a node is equal to or greater than R_m and the node produced at least 1 block in the last cycle, the node's reputation R will be divided by R_r, which is reputation value regulator.

Let the reward and competition cycles be denoted by C_r and C_c respectively. Additionally, let B be the number of blocks produced by a node in a reward cycle while the maximum number of blocks generated by a node with maximum reputation during the competition cycle is B_{max}. The reward E_r for good behavior is then computed as

$$E_r = \frac{B}{\frac{C_r}{C_c} \cdot \frac{R}{R_m} \cdot B_{max}}, \tag{4}$$

if $E_r > 1$, $E_r = 1$, else

$$R = R + \frac{(1 - E_r) \cdot (R_m - R)}{d}, \tag{5}$$

where the divisor d is used for optimization. Users can vary the value of d based on their requirements. Participating nodes with undesirable behavior are penalized during the penalty cycle. Next, let the number of blocks produced by a node and the minimum number of blocks produced by a node with maximum reputation during the penalty cycle, C_p, be denoted by B_p and B_{min} respectively. The penalty, E_p for a node is then computed as

$$E_p = \frac{B_p}{\frac{C_p}{C_c} \cdot \frac{R}{R_m} \cdot B_{min}}, \tag{6}$$

if $E_p > 1$, E_p = 1, else

$$R = R - \frac{(1 - E_p) \cdot (R)}{d}. \tag{7}$$

3) Proof of Contribution: PoC [16] was designed to address the problem of high energy consumption in PoW. To achieve this goal, its designers combine the functionalities of PoW and the energy-efficient component of the PoS consensus protocol which is similar to Peercoin's consensus algorithm. Unlike what is obtainable in Peercoin, where PoW is used generate to blocks first, and when the mining reward decreases, PoS is

used for minting new blocks. The difficulty in Peercoin is adjusted by coinage. PoC is a combination of PoW and behavior staking where the nodes' stake is based on the honesty of the node and not coinage as in Peercoin. The number of times a node adds a valid block to the blockchain is referred to as *success time* while the success time's coefficient determines the mining difficulty (MD) of the miners. This implies that similar to PoRX, each node has a different difficulty level. The higher a node's success time score (STS), the more likely it is for it to be selected as a block producer. It is possible for other nodes to verify the STS and mining difficulty of other nodes by viewing transactions in the blockchain. Also, a blacklist is maintained by nodes to deter misbehavior and to improve the network's resilience to attacks. A node that acts maliciously will be added to the blacklist and have its mining difficulty increased. In addition, all difficulty rewards are revoked.

A difficulty factor, D_f is introduced to properly adjust each node's difficulty according to its success time S_{tm}. If $S_{tm} = 0$, $D_f = 1$ but if the $S_{tm} \neq 0$, then D_f is calculated as

$$D_f = \frac{1}{r + e^{-(S_{tm}-u)/y}}, \tag{8}$$

where r determines the upper bound convergence of the function. The maximum reward for each node can be controlled by varying r. u affects the position of the mean and the value of y determines the growth rate, i.e., higher values of y lead to slower growth. The S_t-scaled difficulty for a node is given as

$$D_n = D \cdot D_f^{-1}, \tag{9}$$

where D, D_f and D_n are the overall network difficulty, difficulty factor in Eq. 8 and the node's new difficulty with respect to S_{tm}. A node will proceed to mine with D_n as its difficulty, with its target calculated using Eq. 1. When a new valid block is added by a node, the latter's success time S_{tm} is incremented and has its difficulty recalculated using Equations 8 and 9.

B. Stochastic Simulation

Stochastic simulation is a model of a system with variables that change stochastically (at random) according to individual probabilities [32]. First, random variables are generated and placed into a system model. After executing the simulation, the model's outputs are recorded and the procedure is repeated with a new set of random values.These steps are repeated until a sufficient amount of data is obtained. The final output distribution gives the most likely estimations as well as a frame of expectations for which value range the random variables are more or less likely to fall into. The random variables are generally instantiated using a random number generator (RNG) [19].

1) Monte Carlo Simulation: MC simulation [33] is one example of stochastic simulation. It is used to model the probability of different outcomes in a process that cannot easily be predicted due to the intervention of random factors. It is a technique used to understand the impact of risk and uncertainty in prediction and forecasting models. Generally, MC assigns random values to variables with uncertainty and runs the model to generate results. This process is repeated with different (random) values assigned to the variable. At the end of the simulation, the results are averaged across all iterations to provide an accurate estimate. MC simulation can be classified into two types known as the sequential (SMC) and non-sequential (NSMC) Monte Carlo. The main difference between the two is the nature of the system states being simulated. NSMC randomizes system states over the duration of the simulation. The simulation is repeated to obtain a statistically reliable average. Results are non-chronological and are produced by simulating each system state separately. In contrast, SMC is chronological in nature whereby the evolution of system states can be observed over a specified duration. The SMC sampling method is based on sampling the probability distribution of the state components. One of the advantages of SMC over NSMC is its ability to model all contingencies and attributes in the system [34], [35]. SMC forms the basis of SIM-P since the selection of block leaders in a consensus protocol has a chronological component, i.e., blocks are produced after a certain duration to allow previous blocks to be broadcasted over the network.

2) Agent-based modeling and Simulation: Agent-based modeling and simulation (ABMS) is an approach for modeling complex systems that are made up of interactive autonomous nodes or agents [36]. These interactive agents are modeled individually and have unique behaviors often defined by rules and their interactions with other agents. These inter-agent interactions can then be observed to identify the emergence of structures, patterns, and behaviors that were not specifically programmed into the models.

Two distinct features of ABMS (as compared to other simulation approaches) include the ability to model heterogeneous agents across a given population and an emergence of self-organization. ABMS allows the modeling of social systems that are made up of agents that interact and influence each other. We designed SIM-P with ABMS as its foundation since the latter can cater to evolving occurrences or emergent phenomena resulting from changes to a node's attributes. Since ABMS is flexible, SIM-P can also be modified with ease to simulate networks of different sizes or other consensus protocols. It also supports the use of a single agent or multiple agents with different levels of description.

IV. The Design of SIM-P

SIM-P is a simplified simulator for consensus protocols based on agent-based modeling and SMC. It is able to accurately simulate the stochastic nature of block discovery without actually having to compute the underlying mechanism such as computing hash values. SIM-P represents each node in the blockchain network as an agent. Probabilistic decisions such as the selection of block leaders are represented as individual trials in a binomial experiment. PoW is used as the base model of the simulator, from which simulation models for other protocols can be developed. Thus PoW will be used as a concrete example of how the SIM-P works for ease of understanding. To decide if a node has successfully found

a solution to the hash puzzle in PoW, a random number is generated and compared to the node's probability of success. If the random number being generated is smaller than or equal to the probability of success, the node is declared the block leader. We will later describe how this simple approach involving random numbers can accurately represent the stochastic process of block leader selection. The probability of success of each participating node is calculated with respect to the mining difficulty, hashing power (number of hashes per second), and the maximum possible target.

Let X be a set of active nodes that participate in the consensus mechanism. These nodes have several attributes that affect their chances of being selected as block leaders. For example, the main attribute for PoW is the hash rate H while other protocols such as PoRX would have reputation, R. The chances of a node being selected to publish a block are determined by a combination of one or more of these attributes, which varies depending on the protocol being simulated. The number of times a simulation will be repeated (trials) and the target are denoted by S_t and T respectively. In each trial, S_t=1 is equivalent to 1 second in real-time. The target is computed with respect to the difficulty of finding a block, D. We assume that the probability of a node winning the right to produce a block follows the binomial distribution. The binomial distribution is used when there are exactly two exclusive outcomes of a trial for a fixed number of independent trials. This accurately describes the block leader selection of a consensus protocol since each attempt to find a block (e.g. finding the correct hash) are independent trials that have two exclusive outcomes (success or failure). The average time it takes a node to find a block and the time it takes a node to find a block are denoted as T_{ra} and T_r respectively. The probability that a node successfully finds a block per trial is

$$P_s^t = \frac{T}{N_t}, \tag{10}$$

where N_t is the number of possible outcomes. The probability of failure per trial is

$$P_f^t = 1 - P_s^t. \tag{11}$$

The probability that a node finds a solution to the hash puzzle based on its hash rate is given as

$$P_s = (1 - P_f^t) \cdot H, \tag{12}$$

where H is the hash rate. Naturally, the overall probability of failure (a node not being able to find a solution) is given as

$$P_f = 1 - P_s. \tag{13}$$

For each trial, a random number r is generated and compared to the probability P_s to determine if the target is met. This simple process can accurately model block leader selection since the latter is commonly a stochastic process that relies on some underlying attribute such as hash rate (PoW) or stake (PoS). Using PoW as an example, the output of the hash function in PoW is assumed to be sampled from a uniformly random distribution. This is a reasonable assumption, considering the fact that double hashing is performed with a good hash function (SHA256). Thus, we can assume that all the 256-bit hash values generated by miners during mining are equiprobable, i.e, they have an equal probability of occurring. Assuming also that the outputs of the random number generator being used in the simulator are being uniformly sampled from the interval of [0,1], setting a threshold within this interval based on P_s is an accurate representation of the stochastic nature of the mining process. For example, if P_s=0.9, we can set the threshold value to be 0.9 to ensure that 90% of the values generated by the random number generator would fall below 0.9. Statistically, a node with $P_s = 10^{-x}$ is then expected to find a solution at least once after 10^x attempts. This can be expressed as

$$Pr(r <= P_s | r \in [0,1]) = \frac{P_s}{1} = P_s. \tag{14}$$

Users can choose to either end the simulation with one of the following methods or set other criteria based on their requirements:

1) Assign a desired value to the simulation time S_t.
2) Calculate the coefficient of variation (COV) for the population sample which is the maximum degree of divergence between the data points. The simulation ends after COV falls under a certain threshold after a specified minimum number of trials, S_{min}. COV is computed as

$$COV = \frac{Std}{mean}, \tag{15}$$

and if $S_t > 1$,

$$COV = \frac{std(T_{ra}(1 : S_t - 1))}{mean(T_{ra}(1 : S_t - 1))}. \tag{16}$$

When using the second ending criteria, the recommended setting for accurate results is $COV = 0.05$ and $S_{min} = 10000$. Figure 1 shows the convergence plot for several experiments using the same set experimental setup, where time (in seconds) refers to the average time required to find a block. This plot shows that when using $COV < 0.05$, the results for all experiments converge after around $S_{min} = 10000$ iterations or trials. We found that these results consistently match theoretical values (See Subsection IV-A). However, users can adopt different values for COV, keeping in mind that the higher the COV, the higher the extent of divergence between the data points while a lower COV leads to longer simulation times. The time record average T_{ra} is computed at the end of the simulation as

$$T_{ra} = \frac{\sum T_r}{S_t}. \tag{17}$$

A summary of the steps involved in each iteration of SIM-P is as follows;

1) Compute P_s (Eq. 12) for each node using their attributes.
2) For each node:
   - Generate r in the range of $[0, 1]$.
   - If $r < P_s$, such node is recorded to have found a block.
3) Check if end criteria have been met, if not, repeat Step 2.
4) Compute the average time taken for each node to succeed.

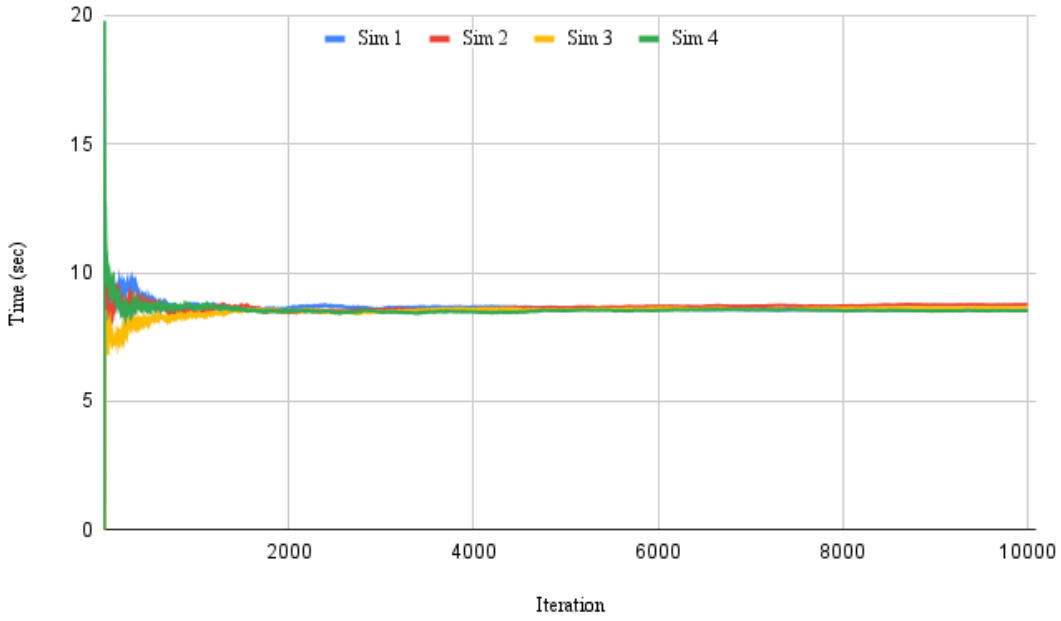


Fig. 1. Monte Carlo simulation convergence

A. Base Proof of Work Model

When it comes to blockchain consensus protocols, PoW is widely used as a benchmark for performance comparison [18] [21] [20]. As such, PoW was selected to be the base model of SIM-P. Each node is assigned its own hash rate H. These "nodes" may not necessarily be individual nodes but can also represent mining pools. The target T of the PoW consensus protocol is then calculated with respect to the global difficulty D using Eq. 1. The probability of success per trial P_s^t for a node is calculated with respect to the target T and N_t possible outcomes as

$$P_s^t = \frac{T}{2^{256}}, \tag{18}$$

where $N_t = 2^{256}$. The probability of failure per trial P_f^t and the overall probability of success P_s with respect to a node's hash rate H are calculated using Eq. 11 and Eq. 12 respectively. The stochastic nature of solving the hash puzzle is simulated by generating a random number r and checking if $r < P_s$. This is performed for each node, every iteration. If the condition is fulfilled, the corresponding node is assumed to have found a solution then T_r and $T_r a$ are recorded. Otherwise, the simulation moves on to the next iteration (the next 1-second time step), and the process repeats. The simulation ends based on one of the end criteria mentioned in Section IV.

1) Verification and Validation of SIM-P Simulation Models: In this section, we verify and validate the SIM-P simulation model from various perspectives [37]. We first verify that the implementation (computational model) of SIM-P is correct by comparing its outputs to a theoretical model before performing data validation using a historical Bitcoin dataset. In addition, internal validity is evaluated through multiple replication runs to ensure that there is minimal variability. This is achieved by calculating the COV across multiple executions.

TABLE III
COMPARISON OF SIM-P AND THEORETICAL RESULTS FOR D=20000, T=13 $\cdot$ 10^{64} AND S_{min} = 10000

Hash rate	SIM-P (hours)	Theoretical result (hours)
10^9	23.81	23.85
10^{10}	2.36	2.38
10^{12}	0.02	0.02

We begin by verifying the correctness of the computational model by comparing its outputs with the theoretical model [38]. We first compute the theoretical estimates for the time required to find a block as a function of the overall network hash rate when the difficulty $D = 20000$. We then performed three separate experiments using a cumulative hash rate of 10^9, 10^{10}, and 10^{12} respectively. The simulation results closely match theoretical estimates as shown in Table III.

Next, we further investigate the validity of the simulation model by comparing its outputs to historical Bitcoin data taken from [39]. The parameters extracted from the latter include hash rate (of mining pools), difficulty, and block discovery time (block time). We used all available samples taken from a 10-year time frame (2011-2020) to ensure that the simulation model is consistent for the entirety of the historical dataset and is unbiased. When simulating higher difficulty values ($> D = 10^6$), a scaling factor, sf needs to be introduced to compensate for computing precision limitations. This scaling factor is used to scale both the hash rate ($\frac{H}{sf}$) and difficulty ($\frac{D}{sf}$). Empirically, we found that selecting sf such that the difficulty is scaled to $< 10^5$ leads to the highly accurate simulations. The results in Table IV show that SIM-P accurately models the behavior of the actual Bitcoin network for all 10 years. The mean absolute error (MAE) of the block time for each year was at most 0.1661 minutes (around 10 seconds). Since the average block

TABLE IV
COMPARISON OF SIM-P RESULTS WITH HISTORICAL BITCOIN DATA SETS BETWEEN JANUARY 2011 TO APRIL 2020. THE MEAN ABSOLUTE ERROR OF THE AVERAGE BLOCK TIME IS COMPUTED IN MINUTES.

Year	Samples	Mean Absolute Error
2011	365	0.1661
2012	366	0.1107
2013	365	0.1386
2014	365	0.1166
2015	365	0.1166
2016	366	0.0995
2017	365	0.1305
2018	365	0.1243
2019	365	0.1160
2020	92	0.1575

time of Bitcoin is around 10 minutes, this translates to around 1.7%.

B. Proof of Reputation-X Model

Next, we show how to extend the base PoW model to simulate other consensus protocols. One straightforward example is PoRX, which includes a reputation module on top of PoW. In general, the reputation module reduces the difficulty for a node as the reputation of the node increases. On top of hash rate H, each node will now have an additional attribute known as reputation R. Recall that maximum reputation, global difficulty, the number of simulations, and local node difficulty are denoted as R_m, D, S_t, and D_l respectively. D_l is calculated from R and D using either Eq. 2 or Eq. 3 depending on whether $R \leq R_m$ or $R > R_m$ respectively.

D_l is then passed to the PoW module where the target T is computed. Therefore, each node will have different target values based on its reputation. The new target values are calculated with respect to D_l using Eq. 1. Thus the probability of a node finding a solution to the hash puzzle is now dependent not only on its hash rate but its reputation. This affects the calculations for the probability of success per trial P_s^t (Eq. 18), probability of failure per trial P_f^t (Eq. 11), overall probability of success P_s (Eq. 12) and failure P_f (Eq. 13).

The rest of the simulation follows PoW with regards to generating the random number r and comparing it to P_s. However, apart from mining fees, a node is rewarded with more reputation if it finds a solution to the hash puzzle. However, if a node fails to do so within a stipulated time, it faces a reputation penalty as described in Subsection III-A2. The reward E_r for good behavior that affects R at the end of a competition cycle is computed using Eq. 4 and Eq. 5 while reputation penalties are calculated using Eq. 6 and Eq. 7. Ending criteria for the PoRX protocol are the same as the ones discussed in Subsection IV-A.

C. Proof of Contribution Model

PoC was designed to solve the high energy consumption problem in PoW by reducing the difficulty of the puzzle to be solved by a node if such node is found to be honest. Honesty is determined by the number of times a node adds a valid block to the blockchain. The higher a node's success time, the easier it will be for the node to find another block. Therefore to simulate PoC, we introduce success time S_{tm} as an additional attribute for each node on top of their hash rate H. Recall that global difficulty and scaled node difficulty (with respect to success time) are denoted as D and D_n respectively. Also, a difficulty factor D_f was introduced to properly adjust each node's difficulty based on S_{tm}. If the $S_{tm} = 0$, $D_f = 1$. Otherwise, D_f is calculated using Eq. 8. D_n for each node can then be calculated using Eq. 9. D_n is then used to calculate the mining target of the underlying PoW protocol using Eq. 1. Calculations for the various success/failure probabilities are the same as PoW and PoRX, taking into consideration the scaled difficulty value. Similarly, the decision of whether a node successfully finds a solution to the hashing problem is performed by comparing random numbers r to P_s. The simulation ends based on one of the criteria in Section IV-A.

In general, PoC rewards nodes for generating valid blocks by reducing their mining difficulty. Each time a node adds a valid block, its corresponding S_{tm} is incremented by 1 and D_n is recomputed based on D_f. Nodes with high success times (which implies a high level of honesty) will have less difficulty finding more blocks.

D. PoRX and PoC Model Validity

Both SIM-P models for PoRX and PoC were first evaluated for internal validity. Multiple replication runs and COV values for both models indicate that they have minimal stochastic variability. Unlike PoW, a comparison to historical datasets cannot be performed for these protocols as they have yet to see widespread adoption. However, under certain conditions, the functionality of these protocols can be reduced to that of PoW. The validation performed for PoW can then be extended to these protocols for these specific instances.

Since PoRX and PoC are derivatives of the PoW protocol, their SIM-P models can be viewed as sub-models of the PoW model with additional attributes that scale mining difficulty. By fixing $E_r = 1$ each round for all nodes, there will be no additional reputation rewarded, which reduces PoRX to the base functionality of PoW. Similarly, if the difficulty factor of PoC is fixed to $D_f = 1$ each round for all nodes, the functionality of PoC is reduced to PoW.

Since the correctness of their underlying mechanism (PoW) was analyzed in Subsection IV-A1, the simulation models for PoRX and PoC (under conditions $E_r = 1$ and $D_f = 1$ respectively) have been equivalently validated. In the absence of real-world data, these models are currently the only alternative that can be used for performance comparison purposes.

V. EXPERIMENTS AND EVALUATION

We implemented SIM-P's base PoW model and proof of concepts models with Python 3.8.8[1] We will show how SIM-P can be used to generate performance metrics for PoW, PoRX, and PoC. These metrics include (but are not limited to) throughput, security against the 51% attack and energy consumption.

[1]SIM-P is publicly available at https://github.com/Oyinloyedp/SIM-P.

TABLE V
THROUGHPUT COMPARISON IN (TPS) FOR POW, PORX AND POC WITH D= 20000 AND S_t=10000

Protocol	5 nodes	10 nodes	100 nodes	1000 nodes
PoW	34.3	55.2	322.5	500
PoRX	72.4	140.8	454.5	526.3
PoC	57.1	90	370.3	526.3

TABLE VI
AVERAGE BLOCK DISCOVERY TIME COMPARISON IN (SECONDS) FOR POW, PORX AND POC WITH D=20000 AND S_t=10000

Protocol	5 nodes	10 nodes	100 nodes	1000 nodes
PoW	29.1	18.1	3.1	2
PoRX	13.8	7.1	2.2	1.9
PoC	17.5	11.1	2.7	1.9

A. Experimental Setup

Throughput: The number of blocks that can be validated and deployed to the blockchain per second is referred to as throughput. Throughput can also be measured in transactions per second, which is the number of processed transactions per unit of time (TPS). The parameters involved in the experiment for measuring the throughput for PoW, PoRX, and PoC are as follows:

1) We first simulate PoW to document the number of blocks produced with the following node attributes and network parameters:
   - Hash rate H_r= (10^8 to 10^{15})
   - Number of simulations $S_t = 10000$
   - Global difficulty $D = 20000$
   - Number of nodes = $\{5, 10, 100, 1000\}$
   - Block size B_s = 1MB
   - Average transaction size T_s = 1KB
2) Simulation for PoRX uses the same parameters as PoW but includes an additional reputation attribute R which is randomly selected between 1000 to 4000. The randomized attribute values simulate the actual scenario of a mature network whereby various nodes would have different reputation levels over time.
3) Simulation for PoC also uses the same parameters as PoW but includes an additional success time attribute S_{tm}. S_{tm} is randomly instantiated with values ranging from 0 to 3. The randomized S_{tm} values depict an actual scenario of a mature network where various nodes would have different success time values over time.
4) Simulation ends when $S_t = 10000$ (first end criteria described in Section IV.
5) SIM-P automatically exports the average time to find a block T_{ra} to a spreadsheet.
6) Compute TPS for all models as

$$(\frac{B_s}{T_s})/T_{ra}. \tag{19}$$

Experimental results for throughput and average block discovery time are shown in Table V and Table VI respectively.

51% Attack: When a node or a pool of nodes amasses 51% of the resources used to determine block leadership, such as computational power or wealth, they can theoretically perform what is known as a 51% attack [40] These node(s) will be able to mine blocks at a faster rate in an attempt to establish a separate chain that is either as long as or longer than the current valid chain. Then, these nodes can perform malicious actions such as falsifying transactions or performing double-spending [41]. To measure a consensus protocol's resilience against the 51% attack, we measure the percentage of blocks discovered by such an attacker per 1 million simulations. The steps and parameters involved in the experiment for measuring the resistance of protocols against the 51% attack are as follows:

1) PoW was first simulated as the base model with the following parameters:
   - Hash rate $H = 10^{15}$ distributed among all nodes with the attacker having exactly 51% of the cumulative hash rate.
   - Number of simulations S_t=10000
   - Global difficulty D =20000
   - Number of nodes = $\{6, 11, 21\}$
   - Unit of measurement = Percentage of blocks found per simulation
2) For PoRX, nodes are assigned a randomly selected reputation R ranging between 1000 and 4000. All other parameters are the same as the PoW experiment.
3) For PoC, nodes are assigned a randomly selected success time S_{tm} ranging between 0 and 3. All other parameters are the same as the PoW experiment.
4) Simulation ends when $S_t = 10000$ (first end criteria described in Section IV.
5) The number of blocks found by the nodes are exported to a spreadsheet.
6) Compute the percentage of blocks discovered by the malicious node with respect to the total number of blocks found.

Experimental results for the 51% attack are recorded in Table VII.

TABLE VII
COMPARISON OF THE PERCENTAGE OF BLOCKS DISCOVERED BY THE 51% ATTACKER INDICATING THE RATE AT WHICH THE ATTACKER FINDS BLOCKS IN POW, PORX AND POC

Protocol	6 nodes	11 nodes	21 nodes
PoW	22.6%	18.6%	16.7%
PoRX	19.6%	11.7%	9.2%
PoC	20.3%	14.7%	12.6%

Energy consumption: As its name implies, energy consumption measures how much energy is expended by a consensus protocol to produce blocks. It is one of the key problems of PoW-based blockchains such as Bitcoin which consumes as much energy as entire countries. The energy consumption of each protocol is measured in kilowatts per transaction (KW/Tx). To calculate energy consumption for each node based on their hash rate, we use the Antminer s19 ASIC miner's power rating as a reference point. The Antminer has a power rating of 3.250 kilowatts per hour (KW/h) and can

TABLE VIII
COMPARISON OF ENERGY CONSUMPTION PER TRANSACTION IN (KW/TX) WITH D=20000 AND S_t=10000

Node ID	5 nodes	10 nodes	100 nodes	1000 nodes
PoW	11.06	11.43	20.03	123.69
PoRX	5.5	5.45	14.63	119.54
PoC	7.04	8.315	17.38	119.54

produce 95 TH/s [42]. Thus, the power rating, E for a node x with a hash rate of H in KW/h is calculated as

$$E_x = \frac{H}{95 \cdot 10^{12}} \cdot 3.250. \tag{20}$$

The steps and parameters involved in the experiment for measuring energy consumption are as follows:

1) The parameters used to simulate PoW are as follows:
   - Hash rate H_r= (10^8 to 10^{15})
   - Number of simulations S_t=10000
   - Global difficulty D =20000
   - Number of nodes as N = 5, 10, 100, 1000
   - Block size = 1MB
   - Average transaction size = 1KB
2) For PoRX, nodes are assigned a randomly selected reputation R ranging between 1000 and 4000. All other parameters are the same as the PoW experiment.
3) For PoC, nodes are assigned a randomly selected success time S_{tm} ranging between 0 and 3. All other parameters are the same as the PoW experiment.
4) Based on the average transactions per second in Table V, the energy consumed per transaction for a set of nodes X is calculated as

$$E_t = \sum_{x \in X} \frac{E_x}{TPS}. \tag{21}$$

The comparison of energy consumption for the three consensus protocols is recorded in Table VIII.

B. Discussion

Throughput: A comparison of the throughput of PoW, PoRX, and PoC is provided in Table V. The TPS for PoW is consistently lower than PoRX and PoC across all simulation categories. When there are fewer nodes (< 10), there is a significant difference between the throughput of PoW as compared to PoRX PoC. Notably, in the 5-node and 10-node scenarios, the TPS of PoRX is double and almost triple that of PoW. Generally, PoC is outperformed by PoRX but still has a higher throughput than PoW (almost 2 times faster). This shows the effectiveness of the reputation module introduced in PoRX in terms of scaling the difficulty level of solving the hash puzzle. The difficulty scaling using reputation in PoRX has a larger impact as compared to scaling based on success time in PoC.

As the number of nodes grows, the performance gap between PoRX and PoC starts to narrow. In fact, all three protocols have a similar average block discovery time when there are 1000 nodes as shown in Table VI. However, this does not imply that the underlying mechanisms in PoRX and PoC are less effective with more nodes but rather the global difficulty of $D = 20000$ is too low given the network's overall cumulative hash rate. Thus blocks are produced at such a high rate that the effect of reputation and success time becomes negligible. Even in this scenario, we still see a slight advantage of using PoRX and PoC over PoW.

51% Attack: Simulation results for the 51% attack are recorded in Table VII. Across all three protocols, an attacker who possesses 51% of the overall network hash rate exhibit dominance over the other nodes, producing blocks at a higher rate than its peers. However, as the number of nodes increases, the dominance of the attacker declines across all simulation categories. The inclusion of reputation and success time has a positive effect on resistance against the 51% attack. An attacker with a similar advantage in terms of hash rate finds fewer blocks in PoRX and PoC as compared to PoW. The difficulty scaling mechanism negates some of the attacker's advantages by reducing the difficulty for other nodes that contribute valid blocks to the network. What this implies is that a malicious node that has accumulated 51% of the resources/power would require more computational effort to launch a successful attack (e.g. double-spending) in PoRX and PoC as compared to PoW.

Energy consumption: All three protocols being analyzed are based on computational effort. The energy consumed per transaction is summarized in Table VIII. Despite all three belonging to the CPE category, PoRX and PoC lead to lower energy consumption due to their underlying difficulty scaling mechanism. This is most noticeable in the experiments involving 5-10 nodes, where the energy consumption, as compared to PoW, is halved for PoRX and is around 40% less for PoC. When the number of nodes is increased to 1000 without increasing the global difficulty, the block production rate starts to saturate due to the high cumulative hash rate. Therefore, a lot of energy is wasted by all three protocols per transaction. However, PoRX and PoC still have a slight advantage over PoW even in this scenario.

PoW relies solely on computational power to select who publishes a block. In contrast, the difficulty scaling mechanism in PoRX and PoC scales the puzzle difficulty of nodes with respect to their reputation and success times. The task of finding the target hash value is made easier and nodes will find a valid solution in a shorter amount of time, which implies that less energy is required to solve the same task. One could argue that PoW can achieve similar energy savings by reducing difficulty. However, both PoRX and PoC can achieve this goal without having to sacrifice other aspects such as resistance against the 51% attack.

VI. Conclusion

In this paper, we proposed SIM-P, a flexible blockchain consensus protocol simulator based on agent-based modeling and the sequential Monte Carlo method. SIM-P represents nodes as agents and the stochastic nature of the block leader selection are represented as independent trials in a binomial experiment. SIM-P provides users with the flexibility to analyze various consensus scenarios and can even possibly be used with optimization algorithms to find optimal performance trade-offs. Since PoW is widely used as a benchmark for blockchain consensus protocol design, it was selected to be the base model for SIM-P. We show that with slight modifications to the base model, other protocols such as PoRX and PoC can be easily simulated. We then show how SIM-P can be used to generate vital performance metrics for consensus protocols such as throughput, resistance to the 51% attack, and energy consumption. Our experiments indicate that, despite its simplicity, SIM-P is able to generate accurate results which we verify using theoretical estimates for PoW and historical Bitcoin data. As a proof of concept and an additional contribution, we compare the performance of PoRX and PoC with PoW based on these metrics and show that the difficulty scaling mechanisms of PoRX and PoC lead to various advantages over the traditional PoW protocol.

Acknowledgment

The full authenticated version of this paper has been published in the IEEE Internet of Things journal and is available at https://doi.org/10.1109/JIOT.2022.3221916. This work was supported by the Universiti Sains Malaysia, Research University Team (RUTeam) Grant Scheme (Grant Number : 1001/PKOMP/8580013) and the Uniten BOLD2025 Research Fund with Project Code J510050002/2021052. The first author, D.P. Oyinloye is funded under the Tertiary Education Trust Fund (TETFUND) Nigeria.

References

[1] B. Bera, S. Saha, A. K. Das, and A. V. Vasilakos, "Designing Blockchain-Based Access Control Protocol in IoT-Enabled Smart-Grid System," *IEEE Internet of Things Journal*, vol. 8, no. 7, pp. 5744–5761, 2021.

[2] Q. Yang and H. Wang, "Privacy-Preserving Transactive Energy Management for IoT-Aided Smart Homes via Blockchain," *IEEE Internet of Things Journal*, vol. 8, no. 14, pp. 11 463–11 475, 2021.

[3] A. Vangala, A. K. Sutrala, A. K. Das, and M. Jo, "Smart Contract-Based Blockchain-Envisioned Authentication Scheme for Smart Farming," *IEEE Internet of Things Journal*, vol. 8, no. 13, pp. 10 792–10 806, 2021.

[4] S. Sharkey and H. Tewari, "Alt-PoW: An Alternative Proof-of-Work Mechanism," in *2019 IEEE International Conference on Decentralized Applications and Infrastructures (DAPPCON)*. IEEE, apr 2019.

[5] D. Puthal and S. P. Mohanty, "Proof of Authentication: IoT-Friendly Blockchains," *IEEE Potentials*, vol. 38, no. 1, pp. 26–29, jan 2019.

[6] Y. Lu, "Blockchain: A Survey on Functions, Applications and Open Issues," *Journal of Industrial Integration and Management*, vol. 03, no. 04, p. 1850015, nov 2018.

[7] P. Ray, B. Chowhan, N. Kumar, and A. Al-Mogren, "BIoTHR: Electronic Health Record Servicing Scheme in IoT-Blockchain Ecosystem," *IEEE Internet of Things Journal*, vol. 8, pp. 10 857–10 872, 07 2021.

[8] G. Sun, M. Dai, F. Zhang, H. Yu, X. Du, and M. Guizani, "Blockchain-Enhanced High-Confidence Energy Sharing in Internet of Electric Vehicles," *IEEE Internet of Things Journal*, vol. 7, no. 9, pp. 7868–7882, 2020.

[9] Z. Su, Y. Wang, Q. Xu, M. Fei, Y.-C. Tian, and N. Zhang, "A secure charging scheme for electric vehicles with smart communities in energy blockchain," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4601–4613, 2019.

[10] C. Liu, K. K. Chai, X. Zhang, and Y. Chen, "Proof-of-Benefit: A Blockchain-Enabled EV Charging Scheme," in *2019 IEEE 89th Vehicular Technology Conference (VTC2019-Spring)*, 2019, pp. 1–6.

[11] J. Zou, B. Ye, L. Qu, Y. Wang, M. A. Orgun, and L. Li, "A Proof-of-Trust Consensus Protocol for Enhancing Accountability in Crowdsourcing Services," *IEEE Transactions on Services Computing*, vol. 12, no. 3, pp. 429–445, may 2019.

[12] J. M. Kim, J. Won Lee, K. Lee, and J. Huh, "Proof of phone: A low-cost blockchain platform," in *2019 IEEE International Conference on Consumer Electronics (ICCE)*, 2019, pp. 1–4.

[13] F. Gai, B. Wang, W. Deng, and W. Peng, "Proof of Reputation: A Reputation-Based Consensus Protocol for Peer-to-Peer Network," in *Database Systems for Advanced Applications*. Springer International Publishing, 2018, pp. 666–681.

[14] E. K. Wang, Z. Liang, C.-M. Chen, S. Kumari, and M. K. Khan, "PoRX: A Reputation Incentive Scheme for Blockchain Consensus of IIoT," *Future Generation Computer Systems*, vol. 102, pp. 140–151, jan 2020.

[15] T. Ogawa, H. Kima, and N. Miyaho, "Proposal of Proof-of-Lucky-Id(PoL) to Solve the Problems of PoW and PoS," in *2018 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData)*. IEEE, jul 2018.

[16] T. Xue, Y. Yuan, Z. Ahmed, K. Moniz, G. Cao, and C. Wang, "Proof of Contribution: A Modification of Proof of Work to Increase Mining Efficiency," *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, vol. 01, pp. 636–644, 2018.

[17] D. P. Oyinloye, J. S. Teh, N. Jamil, and M. Alawida, "Blockchain Consensus: An Overview of Alternative Protocols," *Symmetry*, vol. 13, p. 1363, 2021.

[18] M. Alharby and A. van Moorsel, "BlockSim: A Simulation Framework for Blockchain Systems," *SIGMETRICS Perform. Eval. Rev.*, vol. 46, no. 3, p. 135–138, jan 2019.

[19] C. Faria and M. P. Correia, "BlockSim: Blockchain Simulator," *2019 IEEE International Conference on Blockchain (Blockchain)*, pp. 439–446, 2019.

[20] L. Stoykov, K. Zhang, and H.-a. Jacobsen, "VIBES: Fast Blockchain Simulations for Large-Scale Peer-to-Peer Networks: Demo," in *Proceedings of the 18th ACM/IFIP/USENIX Middleware Conference: Posters and Demos, Las Vegas, NV, USA, December 11 - 15, 2017*, 12 2017, pp. 19–20.

[21] A. Gervais, G. O. Karame, K. Wüst, V. Glykantzis, H. Ritzdorf, and S. Capkun, "On the Security and Performance of Proof of Work Blockchains," *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

[22] "Bitcoin simulator," https://www.nsnam.org/tutorials/NS-3-LABMEETING-1.pdf/, accessed: 2021-12-02.

[23] T. T. A. Dinh, J. Wang, G. Chen, R. Liu, B. C. Ooi, and K.-L. Tan, "BLOCKBENCH: A Framework for Analyzing Private Blockchains," *Proceedings of the 2017 ACM International Conference on Management of Data*, 2017.

[24] X. Wang, A. Al-Mamun, F. Yan, M. Sadoghi, and D. Zhao, "BlockLite: A Lightweight Emulator for Public Blockchains," *ArXiv*, vol. abs/1905.02157, 2019.

[25] Y. Aoki, K. Otsuki, T. Kaneko, R. Banno, and K. Shudo, "Simblock: A blockchain network simulator," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 325–329.

[26] M. Zander, T. Waite, and D. Harz, "DAGsim: Simulation of DAG-based distributed ledger protocols," *ACM SIGMETRICS Performance Evaluation Review*, vol. 46, pp. 118–121, 01 2019.

[27] J. Yang, M. Onik, N.-Y. Lee, M. Ahmed, and C.-S. Kim, "Proof-of-Familiarity: A Privacy-Preserved Blockchain Scheme for Collaborative Medical Decision-Making," *Applied Sciences*, vol. 9, no. 7, p. 1370, apr 2019.

[28] E. K. Wang, R. Sun, C.-M. Chen, Z. Liang, S. Kumari, and M. K. Khan, "Proof of X-repute blockchain consensus protocol for IoT systems," *Computers & Security*, vol. 95, p. 101871, aug 2020.

[29] A. Berentsen, "Aleksander Berentsen Recommends "Bitcoin: A Peer-to-Peer Electronic Cash System" by Satoshi Nakamoto," in *21st Century Economics*. Springer International Publishing, 2019, pp. 7–8.

[30] H. F. Ouattara, D. Ahmat, F. T. Ouédraogo, T. F. Bissyandé, and O. Sié, "Blockchain Consensus Protocols," in *Lecture Notes of the Institute*

for Computer Sciences, Social Informatics and Telecommunications Engineering. Springer International Publishing, 2018, pp. 304–314.
[31] N. Chaudhry and M. M. Yousaf, "Consensus Algorithms in Blockchain: Comparative Analysis, Challenges and Opportunities," in *2018 12th International Conference on Open Source Systems and Technologies (ICOSST)*. IEEE, dec 2018.
[32] P.-Y. Piriou and J.-F. Dumas, "Simulation of stochastic blockchain models," in *2018 14th European Dependable Computing Conference (EDCC)*, 2018, pp. 150–157.
[33] J. Menčík, "Monte Carlo Simulation Method," in *Concise Reliability for Engineers*, J. Mencik, Ed. Rijeka: IntechOpen, 2016, ch. 15.
[34] M. G. Cox and B. R. L. Siebert, "The Use of a Monte Carlo Method for Evaluating Uncertainty and Expanded Uncertainty," *Metrologia*, vol. 43, no. 4, pp. S178–S188, aug 2006.
[35] D. Frenkel, "Introduction to Monte Carlo Methods," *J. von Neumann Inst. Comput.*, vol. 23, 01 2004.
[36] C. Macal and M. North, "Tutorial on agent-based modelling and simulation," *J. Simulation*, vol. 4, pp. 151–162, 09 2010.
[37] R. G. Sargent, "Verification and validation of simulation models," in *WSC*. WSC, 2007, pp. 124–137.
[38] "Difficulty," https://en.bitcoin.it/wiki/Difficulty, accessed: 2021-12-01.
[39] "Bitcoin historical data," https://data.bitcoinity.org/markets/volume/30d?c=e&t=b/, accessed: 2021-05-08.
[40] S. Sayeed and H. Marco-Gisbert, "Assessing Blockchain Consensus and Security Mechanisms against the 51% Attack," *Applied Sciences*, vol. 9, no. 9, p. 1788, apr 2019.
[41] B. Hamdaoui, M. Alkalbani, A. Rayes, and N. Zorba, "IoTShare: A Blockchain-Enabled IoT Resource Sharing On-Demand Protocol for Smart City Situation-Awareness Applications," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 548–10 561, 2020.
[42] H. Jung and H.-N. Lee, "ECCPoW: Error-Correction Code based Proof-of-Work for ASIC Resistance," *Symmetry*, vol. 12, p. 988, 06 2020.



Damilare Peter Oyinloye received his B.Sc. (Hons.) and M.Sc. degrees in computer science from Kwara State University, Malete, Nigeria, in 2013 and 2016, respectively. He is currently pursuing a Ph.D. degree at the School of Computer Sciences, Universiti Sains Malaysia. He is also a Lecturer and Researcher with Kwara State University,Nigeria. His research interests include blockchain technology, information security, cryptography and IoT.



Je Sen Teh received his Ph.D. in Computer Science from Universiti Sains Malaysia 2017. He is currently a Senior Lecturer at Universiti Sains Malaysia. His research interests include blockchain technology, and the design and cryptanalysis of symmetric-key cryptographic algorithms.



Norziana Jamil received her PhD in Security in Computing (Cryptographic Hash Functions) in 2013. She is now serving Universiti Tenaga Nasional as an Associate Professor. Her area of specialization and interest includes Cryptography, security for Cyber-Physical Systems, security analytics and intelligence.



Jiashen Teh received his Ph.D degree in electrical and electronic engineering from the University of Manchester in 2016. He is now a Senior Lecturer at Universiti Sains Malaysia. He is also an Adjunct Professor at the Intelligent Electric Vehicle & Green Energy Center of National Chung Hsing University and is the Technical Director of UPE-Power co. ltd., Taichung, Taiwan. His research interests include probabilistic modelling of power systems, grid-integration of renewable energy sources and reliability modelling of smart grid networks.

View publication stats